Amendments to the Claims:

1. (Currently amended) A method for representing header and footer structures in a markup language document, comprising:

determining properties corresponding to a mini-document that relates to at least one section of an application document~~, wherein the properties comprise at least one of a context free chunk element and a table element~~;

mapping the properties of the mini-document into a markup language element, wherein mapping includes mapping the properties into at least one member of a group comprising: a context free chunk element and a table element; and

storing the properties of the mini-document in the markup language document.

2. (Original) The method of Claim 1, further comprising determining whether the mini-document is one of a header and a footer.

3. (Original) The method of Claim 1, wherein mapping the properties further comprises mapping a type attribute that corresponds to the mini-document.

4. (Original) The method of Claim 3, wherein the type attribute corresponds to whether the mini-document occurs on a first page, odd pages, or even pages of the specified section of the application document.

5. (Cancelled)

6. (Currently amended) The method of Claim 1, further comprising:

determining properties corresponding to an additional mini-document that relates to at least one section of the application document~~, wherein the properties comprise at least one of a context free chunk element and a table element~~;

mapping the properties of the additional mini-document into a markup language element, wherein mapping includes mapping the properties into at least one member of a group comprising: a context free chunk element and a table element; and

storing the properties of the additional mini-document in the markup language document.

7. (Original) The method of Claim 1, further comprising:

determining whether properties associated with all mini-documents of the application document have been stored in the markup language document; and

processing further mini-documents when the properties associated with all mini-documents have not been stored in the markup language document.

8. (Original) The method of Claim 1, wherein the properties of the mini-document stored in the markup language document are understood by an application that understands the markup language when the mini-document is not native to the application.

9. (Original) The method of Claim 1, wherein the markup language document is manipulated on a server to substantially reproduce the mini-document of the application document notwithstanding the presence of an application that generated the markup language document.

10. (Currently amended) A computer-readable medium for representing headers and footers in a markup language document, comprising:

determining properties relating to a mini-document used within a word-processing document~~, wherein the properties comprise at least one of a context free chunk element and a table element~~;

determining whether the mini-document is one of a header and a footer;

writing the properties into a markup language element, wherein writing includes writing the properties into at least one member of a group comprising: a context free chunk element and a table element; and storing the properties in the markup language document such that the headers and footers of the word-processing document are substantially maintained when the markup language document is parsed by an application.

11. (Original) The computer-readable medium of Claim 10, wherein the markup language document is manipulated on a server to substantially reproduce the mini-document of the word-processing document notwithstanding the presence of an application that generated the markup language document.

12. (Original) The computer-readable medium of Claim 10, wherein the properties of the mini-document stored in the markup language document are understood by an application that understands the markup language when the mini-document is not native to the application.

13. (Original) The computer-readable medium of Claim 10, wherein mapping the properties further comprises mapping a type attribute that corresponds to the mini-document.

14. (Original) The computer-readable medium of Claim 13, wherein the type attribute corresponds to whether the mini-document occurs on a first page, odd pages, or even pages of the specified section of the word-processing document.

15. (Cancelled)

16. (Currently amended) The computer-readable medium of Claim 10, further comprising:

determining properties corresponding to an additional mini-document that relates to at least one section of the word-processing document~~, wherein the properties comprise at least one of a context free chunk element and a table element~~;

mapping the properties of the additional mini-document into a markup language element, wherein mapping includes mapping the properties into at least one member of a group comprising: a context free chunk element and a table element; and storing the properties of the additional mini-document in the markup language document.

17. (Original) The computer-readable medium of Claim 10, further comprising:

determining whether properties associated with all mini-documents of the word-processing document have been stored in the markup language document; and

processing further mini-documents when the properties associated with all mini-documents have not been stored in the markup language document.

18. (Currently amended) A system for representing header and footer information in a markup language document, comprising:

a processor; and

a memory associated with computer-executable instructions configured to:

~~an application that is configured to:~~

determine properties relating to a mini-document included in at least one section of an application document~~, wherein the properties comprise at least one of a context free chunk element and a table element~~;

determine whether the mini-document is one of a header and a footer;

map the properties into a markup language element, wherein mapping includes mapping the properties into at least one member of a group comprising: a context free chunk element and a table element; and

store the properties in the markup language document; and

a validation engine configured to validate the markup language document.

19. (Currently amended) The system of Claim 18, wherein the application is further configured to:

determine properties corresponding to an additional mini-document that relates to at least one section of the application document~~, wherein the properties comprise at least one of a context free chunk element and a table element~~;

map the properties of the additional mini-document into a markup language element, wherein mapping includes mapping the properties into at least one member of a group comprising: a context free chunk element and a table element; and

store the properties of the additional mini-document in the markup language document.

20. (Original) The system of Claim 18, wherein the application is further configured to:

determine whether properties associated with all mini-documents of the application document have been stored in the markup language document; and

process further mini-documents when the properties associated with all mini-documents have not been stored in the markup language document.

21. (Original) The system of Claim 18, wherein the properties of the mini-document stored in the markup language document are understood by an additional application that understands the markup language when the mini-document is not native to the additional application.

22. (Original) The system of Claim 18, wherein the markup language document is manipulated on a server to substantially reproduce the mini-document of the application document notwithstanding the presence of the application that generated the markup language document.